III. REMARKS

1. Claims 1, 2, 4-7, 9-16, 18-24, 26-30, 32-38, 40-43, 45-47, 49-58, 61, and 62 remain in the application. Claims 3, 8, 17, 25, 31, 39, 44, 48, 59, 60, and 63 have been cancelled without prejudice.

2. Applicants respectfully submit that claims 1, 2, 4-7, 9-16, 18-24, 26-30, 32-38, 40-43, 45-47, 49-58, 61 and 62 are patentable over the combination of Bahrs et al. (US 7,181,686, "Bahrs") and Hatanaka et al. (US 5,926,177, "Hatanaka") under 35 USC 103(a).

The combination of Bahrs and Hatanaka fails to disclose or suggest the features of:

> *determining a view chain data structure comprising at least three entries, each of said entries comprising an application identifier and a view identifier, a view identified by said view identifier being within an application identified by said application identifier; passing said view chain data structure to a view router from a first application; detecting a first entry in said view chain data structure by said view router; determining a first target application identifier in said first entry by said view router; launching a first view in said first target application by calling a method associated with said first target application by said view router, said launching comprising a presentation of a first user interface form in said computer graphical user interface by said first target application; receiving data to the first view from a user during said launching of the first view; continuing said view router by calling a listener method associated with said view router by said first target application; checking whether entries for views not launched remain in said view chain data structure, each said entry for a view not launched specifying a view identifier for a view not yet launched; detecting a second entry in said view chain data structure by said view router; determining a second target application identifier in said second entry by said view router; launching a second view in said second target application by calling a method associated with said second target application by said view router when entries for views not launched remain in said view chain, said launching comprising a presentation of a second user interface form in said computer graphical user interface by said second target application; receiving data to the second view from the user during said launching of the second view; continuing said view router by calling a listener method associated with said view router by said second target application; continuing*

said first application automatically when no entries for views not launched remain in said view chain data structure by calling a listener method associated with said first application by said view router,

as recited by independent claims 1, 15, 29 and 43.

Bahrs discloses an architectural model for presenting a graphical user interface in a data processing system. The graphical user interface may be typically for an administrative information system using a remote database. The architecture represents a variation of the model-view-controller approach of designing user interface applications, typically in Windows or Java applications. The architectural model comprises as its main components a view controller, an application mediator, a transporter and a number of destinations. There is also a placement listener, a top listener and validation rules. The view controller represents an actual tangible panel, which in Windows parlance is often referred to as a window. A panel comprises a number of user interface components such as data entry fields and boxes, scrollers, radio buttons and function buttons, in other words widgets (Bahrs, column 31, lines 58 – 67). A single application may comprise multiple panels for different use cases, that is, logically separate man-machine interaction sequences for a particular purpose such as employer contact information enquiry in a large multi-purpose personnel information system.

The view controller is thus a panel that contains a single reusable screen of user display and input. The view controller receives user actions pertaining to the components, such as selections of buttons with the mouse, in the form of an Abstact Window Toolkit Event (AWTEvent) from the operating system. The AWTEvent is used to form a view event, which is provided to the application mediator for processing from the view controller. There is a view controller for each panel in the application. An application mediator is in charge of one or many view controllers. There is an application mediator instance for each use case within an application (Bahrs, column 33, lines 12 – 32). The application mediator communicates with a placement listener though placement events. Placement listener performs the actual placing of panels on different positions on the display, on the basis of instructions from the application mediator, which determines the ordering of view controller associated panels (Bahrs, column 16, lines 22 – 31). The placement performed is in response to view events from the view controller. The application mediator formats request events pertaining to data to be sent or received.

The request events are sent to a transporter, which is responsible for mapping request events to correct destinations. The Transporter acts as an event broadcaster to route request events to the appropriate destination. The transporter acts as a message queue between the application mediator and a number of destinations (Bahrs, column 43, rows 3 – 23). Typically, request events received from the application mediator are requests to obtain data from a database interfaced by a destination, whereas request events received from a destination are result data from a database interfaced by that particular destination. The destination processes the request events by translating data between different formats and interfaces various external protocols for remote database access (Bahrs, Figure 68). By a destination is meant in Bahrs remote or local data storage.

Hatanaka discloses a user interface that provides multiple views to same data. The multiple views are provided from a single model using a controller and a view proxy. The views describe different aspects of the same data such as functional and real or textual and graphical. Particularly, Hatanaka discusses the presentation of network management information. Associated with a hosts or communication links there is status information regarding whether they are operational or not. Hosts and links are represented in the model using objects. The object properties relate, for example, to state of invocation, number of invocations, bytes transferred, maximum transfer rate and time since last activity.

Bahrs fails to disclose or suggest the feature of determining a view chain data structure comprising at least three entries, each of said entries comprising an application identifier and a view identifier, a view identified by said view identifier being within an application identified by said application identifier. In Bahrs there is nothing equivalent to a view chain data structure that comprises entries that further comprise application identifiers. In Bahrs the application mediator does create one or more view controllers, which in turn create and display the panels associated with them. However, the view controllers are not applications. The term "application" has an established meaning for a person skilled in the art and it may not be interpreted as a separate object class.

Bahrs also fails to disclose or suggest the feature of passing said view chain data structure to a view router from a first application. The passage cited by the Examiner in column 3, lines 24 – 26 relates merely to the fact that the application mediator may determine a need for a service and that an event is generated in response to the need. The event is passed to a listener

method in order to perform the service. The cited passage in column 3, 24 – 26 just teaches the very basic principle of message passing in all object oriented programming. As evident from the overview of the cited art above, the event may relate to anything, it may be related to any kind of user input and may be passed to any of the object classes with which the application mediator communicates such as top listener, placement listener, transporter or view controller. Further, the Examiner cites column 16, lines 60 – 67 and column 17, lines 1 – 5 which are an unclear choice by the Examiner because they describe the function of the transporter class, which has nothing to do with user interaction. Instead, it deals with request routing towards various data storage systems. The transporter class is responsible for mapping request events to correct destinations. The Transporter acts as an event broadcaster to route request events to the appropriate destination. The transporter acts as a message queue between the application mediator and a number of destinations (Bahrs, column 43, rows 3 – 23). Typically, request events received from the application mediator are requests to obtain data from a database interfaced by a destination, whereas request events received from a destination are result data from a database interfaced by that particular destination. The destination processes the request events by translating data between different formats and interfaces various external protocols for remote database access (Bahrs, Figure 68). By a destination is meant in Bahrs remote or local data storage. Therefore, Bahrs does not disclose anything that suggests passing a view chain data structure to a view router from a first application.

Bahrs further fails to disclose or suggest the feature of detecting a first entry in said view chain data structure by said view router. As discussed above in Bahrs there is no identifiable view chain data structure which that is processed by a view router that has obtained it from a first application.

In addition, Bahrs fails to disclose or suggest the feature of determining a first target application identifier in said first entry by said view router.

Still further, Bahrs fails to disclose or suggest the feature of launching a first view in said first target application by calling a method associated with said first target application by said view router, said launching comprising a presentation of a first user interface form in said computer graphical user interface by said first target application. In Bahrs there is no teaching regarding a possibility to launch views from a separate target application. For example, Figure 5 in Bahrs does not depict an interface to a target application separate from the illustrated application 500

and its components 502 – 538. In Bahrs there is no discussion of the classes to be used in such a case.

Bahrs also fails to disclose or suggest launching a second view in said second target application by calling a method associated with said second target application by said view router when entries for views not launched remain in said view chain, said launching comprising a presentation of a second user interface form in said computer graphical user interface by said second target application.

There is also no disclosure or suggestion in Bahrs related to continuing said first application automatically when no entries for views not launched remain in said view chain data structure by calling a listener method associated with said first application by said view router. Column 16, lines 22 – 35 cited by the Examiner fail to disclose that a data structure is inspected for views not launched and that a first application is continued in response. In fact it is unclear how this passage is associated with the feature of continuing said first application automatically when no entries for views not launched remain in said view chain data structure by calling a listener method associated with said first application by said view router.

Turning now to Hatanaka, this reference fails to disclose or suggest the feature of checking whether entries for views not launched remain in said view chain data structure, each said entry for a view not launched specifying a view identifier for a view not yet launched. Column 5, lines 1 – 15 in Hatanaka discusses state information associated with objects that represent managed network entities such as data links, hosts and paths and their current attributes such as bytes transferred, time since last activity and state of invocation. However, the indicated passage in Hatanaka merely teaches that object attribute values regarding network components may change from time to time. This is quite different from teaching the feature of checking whether entries for views not launched remain in said view chain data structure, each said entry for a view not launched specifying a view identifier for a view not yet launched. In Hatanaka there is no mentioning of a view chain data structure, the launching of views or view launched or not yet launched. Therefore, clearly Hatanaka fails to disclose the feature of checking whether entries for views not launched remain in said view chain data structure, each said entry for a view not launched specifying a view identifier for a view not yet launched.

Because none of the references disclose or suggest these features, the cited combination fails to disclose or suggest all the features of claims1, 15, 29 and 43.

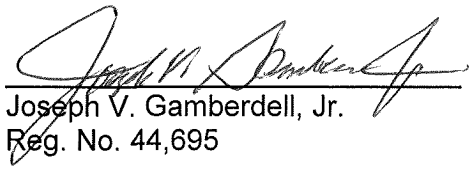
In light of the discussion above, a person skilled in the art would not know how to arrive to the invention as claimed in independent claims 1, 15, 29 and 43 on the basis of the teachings provided by the combination of Bahrs and Hatanaka.

For all the foregoing reasons, it is respectfully submitted that the combination of Bahrs and Hatanaka fails to disclose or suggest all the features of independent claims 1, 15, 29 and 43, and therefore, the combination of Bahrs and Hatanaka fails to render claims 1, 2, 4-7, 9-16, 18-24, 26-30, 32-38, 40-43, 45-47, 49-58, 61 and 62 unpatentable.

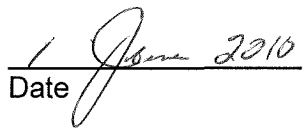
For all of the foregoing reasons, it is respectfully submitted that all of the claims now present in the application are clearly novel and patentable over the prior art of record, and are in proper form for allowance. Accordingly, favorable reconsideration and allowance is respectfully requested. Should any unresolved issues remain, the Examiner is invited to call Applicants' attorney at the telephone number indicated below.

The Commissioner is hereby authorized to charge payment for any fees associated with this communication or credit any over payment to Deposit Account No. 16-1350.

Respectfully submitted,



Joseph V. Gamberdell, Jr.
Reg. No. 44,695



Date

Perman & Green, LLP
99 Hawley Lane
Stratford, CT 06614
(203) 259-1800
Customer No.: 2512